

Рисование геометрических фигур с помощью Turtle на языке Python

Кокухина Софья Александровна

Приамурский государственный университет им.Шолом-Алейхема

Студент

Аннотация

Цель статьи — разработка программы на языке Python для рисования геометрических фигур с помощью библиотеки turtle. Реализация проекта позволяет пошагово выводить на экран основные фигуры (квадрат, треугольник, круг, звезду) с использованием встроенных средств графики Python. Работа демонстрирует основы процедурного программирования, работу с циклами, функциями и модулями, а также принципы управления «черепашкой» в графической среде. Проект выполнен в среде разработки PyCharm, что позволяет наглядно изучить и применять библиотеку turtle для образовательных целей.

Ключевые слова: Python, turtle, PyCharm, графика, геометрические фигуры.

Drawing geometric shapes using Turtle in Python

Kokukhina Sofya Alexandrovna

Sholom-Aleichem Priamursky State University

Student

Abstract

This article presents the development of a Python program for drawing geometric shapes using the turtle library. The implementation demonstrates step-by-step rendering of basic shapes (square, triangle, circle, star) using Python's built-in graphics tools. The work illustrates procedural programming basics, usage of loops, functions, and modules, as well as controlling the turtle in a graphical environment. The project was developed in the PyCharm IDE, which provides a clear visualization and educational utility of the turtle library.

Keywords: Python, turtle, PyCharm, graphics, geometric shapes.

1 Введение

1.1 Актуальность

Использование языка Python для обучения программированию становится всё более популярным благодаря его простоте и широкому спектру библиотек. Одной из таких библиотек является turtle, позволяющая рисовать графические элементы, управляя курсором в виде черепахи. Проекты с использованием turtle позволяют визуализировать алгоритмы, что делает процесс изучения более наглядным и интерактивным. Реализация программы для рисования геометрических фигур даёт возможность студентам освоить

базовые конструкции Python — циклы, функции, условия и применить их в творческом контексте.

1.2 Обзор исследований

В статье М.Н. Подоксенова рассматривается концепция подготовки проведения вебинаров по математике в рамках подготовки школьников к централизованному тестированию, особое внимание уделяется школьникам 9, 10 и 11 классов, предоставляя им возможность получить качественную подготовку к централизованному тестированию перед поступлением в высшие и средние специальные учебные заведения [1]. В публикации О. В Камозиной и О.В. Охлупиной описаны технические средства, на которых базируется создание атрибутов веб-приложения, и указаны требования к программному обеспечению, необходимому для использования ресурса. Определён доступ к частям приложения в зависимости от статуса пользователя. Разработана подробная инструкция для корректной работы школьников с приложением. [2]. В статье Н.А. Стукаловой говорится о повышении качества математической подготовки ориентированных на обучение в вузе старшеклассников в системе дополнительного образования [3]. Н.В. Кашина рассказывает о популярных технологиях создания электронных учебных пособий [4].

1.3 Цель исследования

Цель исследования – разработать единый проект в PyCharm на Python 3.x, реализующий отрисовку набора геометрических фигур (правильные многоугольники, круг, звезда) с показом алгоритмов, параметризацией и возможностью интерактивного управления (через клавиши).

2 Материалы и методы

Для реализации проекта используется язык: Python 3.10 – 3.12 (рекомендуется 3.12).

Библиотека: встроенный модуль turtle (входит в стандартную библиотеку Python; использует Tkinter).

Среда разработки: PyCharm Community Edition (создание проекта, настройка интерпретатора, отображение консоли и графического окна).

ОС: Windows / Linux / macOS (для Linux может потребоваться установить пакет python3-tk). Методы: процедурное программирование, функции, циклы, работа с графическим окном.

3 Результаты

Можно переходить к объявлению и инициализации основных переменных для реализации программы по рисованию геометрических фигур с помощью библиотеки turtle. Создаётся объект черепахи, переменные для размеров окна, цветов фона и пера, а также скорость отрисовки. Дополнительно инициализируется объект экрана, на котором будут отображаться все фигуры (Рис. 1).

```
1  # main.py
2  import turtle
3  import math
4
5  # --- Настройка окна и черепахи ---
6  SCREEN_WIDTH, SCREEN_HEIGHT = 900, 700
7  BG_COLOR = "white"
8  PEN_SIZE = 2
9  DEFAULT_SPEED = 4
10
```

Рисунок 1 - Объявление переменных и инициализация объектов

Далее реализуется функция инициализации рабочего окна и черепахи. В этой функции настраивается размер окна, цвет фона, заголовок, толщина пера и скорость перемещения курсора. Здесь же определяется начальное положение черепахи в центре координатной плоскости (Рис. 2).

```
11 screen = turtle.Screen()
12 screen.setup(SCREEN_WIDTH, SCREEN_HEIGHT)
13 screen.title("Turtle - Geometric Shapes")
14 screen.bgcolor(BG_COLOR)
15
16 t = turtle.Turtle()
17 t.pensize(PEN_SIZE)
18 t.speed(DEFAULT_SPEED)
19 t.hideturtle() # скрываем курсор для чистой отрисовки
20
```

Рисунок 2 – Настройка графического окна и параметров черепахи

Логика программы строится на наборе функций для рисования отдельных фигур: квадрата, треугольника, круга, звезды и произвольного многоугольника (Рис. 3). Каждая функция принимает параметры длины стороны или радиуса, цвета линий и заливки, а также начальных координат. Это позволяет гибко задавать внешний вид и расположение объектов на экране.

```

28 def clear_and_center():
29     """Очищает холст и возвращает черепаху в центр."""
30     t.clear()
31     move_to(x: 0, y: 0)
32     t.setheading(0)
33
34 # --- Базовые примитивы ---
35 def draw_regular_polygon(n, side, start_pos=(0,0), rotate=0, pen_color="black", fill=None):
36     """Рисует правильный n-угольник со стороной side, начиная в start_pos."""
37     if n < 3:
38         raise ValueError("n должно быть >= 3")
39     move_to(*start_pos)
40     t.setheading(rotate)
41     t.pencolor(pen_color)
42     if fill:
43         t.fillcolor(fill)
44         t.begin_fill()
45     exterior = 360.0 / n
46     for _ in range(n):
47         t.forward(side)
48         t.left(exterior)
49     if fill:
50         t.end_fill()
51
52 def draw_square(side, start_pos=(0,0), **kwargs):
53     draw_regular_polygon(n: 4, side, start_pos=start_pos, **kwargs)

```

Рисунок 3 – Пример функции рисования квадрата

Основная функция запуска (draw_demo) вызывает подготовленные функции в определённой последовательности, размещая фигуры в разных частях окна. Для точного позиционирования используется вспомогательная функция move_to, которая перемещает черепаху в указанную точку без рисования линии. Это гарантирует отсутствие пересечений между фигурами и позволяет формировать аккуратную композицию (Рис. 4).

```

def draw_demo():
    clear_and_center()
    # квадрат, треугольник и круг на верхней строке
    draw_square(side: 120, start_pos=(-320, 200), pen_color="green", fill="lightgreen")
    draw_triangle(side: 140, start_pos=(-120, 200), pen_color="blue", fill="lightblue")
    draw_circle(radius: 60, start_pos=(120, 200), pen_color="red", fill="pink")
    # звезда и шестиугольник внизу
    draw_star(size: 140, start_pos=(-240, -50), pen_color="purple", fill="violet")
    draw_regular_polygon(n: 6, side: 70, start_pos=(80, -70), pen_color="orange", fill="yellow")

```

Рисунок 4 – Метод демонстрационного построения фигур

Для управления программой используется обработка нажатий клавиш с помощью методов onkey и listen (Рис. 5). Пользователь может очистить холст, нарисовать демонстрационный набор фигур или вызвать отрисовку конкретной фигуры (например, клавиша 1 — квадрат, 2 — треугольник, 3 — круг, 4 — звезда). Также добавлена возможность сохранить рисунок в файл формата EPS для дальнейшей конвертации в изображение.

```
# --- Привязка клавиш (интерактивность) ---  
def bind_keys():  
    screen.onkey(clear_and_center, key="c") # очистить  
    screen.onkey(draw_demo, key="d") # демонстрация (d)  
    screen.onkey(lambda: draw_square(side: 80, start_pos=(250, -200), pen_color="black"), key="1")  
    screen.onkey(lambda: draw_triangle(side: 80, start_pos=(250, -200), pen_color="black"), key="2")  
    screen.onkey(lambda: draw_circle(radius: 40, start_pos=(250, -200), pen_color="black"), key="3")  
    screen.onkey(lambda: draw_star(size: 90, start_pos=(250, -200), pen_color="black"), key="4")  
    screen.onkey(lambda: save_canvas("output.eps"), key="s") # сохранить (s)  
    screen.listen()
```

Рисунок 5 – Настройка обработчиков событий клавиатуры

После завершения кода можно нажать Shift + F10 в PyCharm для запуска программы. В открывшемся окне будут отображаться все фигуры из демонстрационной функции (Рис. 6). При нажатии соответствующих клавиш пользователь может добавлять новые фигуры в разные точки окна или очищать холст.

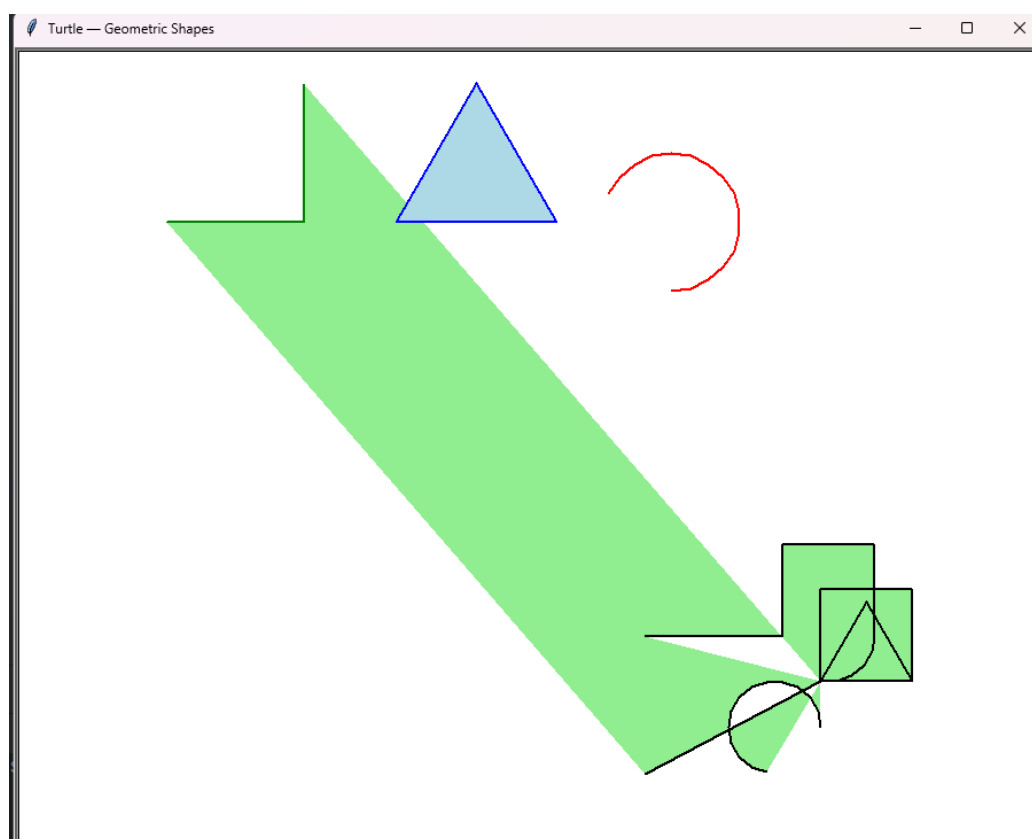


Рисунок 6 – Интерфейс программы с набором геометрических фигур

В процессе работы программа позволяет быстро изменять параметры фигур и их расположение, что делает проект полезным для изучения алгоритмов рисования, координатной геометрии и работы с событиями в Python. Интерфейс остаётся интуитивным и наглядным: чёткие линии, заливка контрастными цветами, простое управление через клавиатуру.

4

Выводы. Реализация единого проекта для рисования геометрических фигур с помощью turtle показала, что библиотека turtle даёт простой и наглядный API для демонстрации алгоритмов рисования. Параметризация функций (количество сторон, длина стороны, цвет, положение) делает код переиспользуемым и расширяемым. Также использование вспомогательных функций (move_to, clear_and_center) упрощает организацию холста и предотвращает случайные наложения. Привязка к клавишам обеспечивает базовый интерактивный интерфейс без сложных GUI-библиотек — полезно для учебных демонстраций.

Для подготовки иллюстраций к статье удобно использовать canvas.postscript() с последующей конвертацией в PNG/ JPG — это позволяет вставлять качественные изображения в Word/Docx.

Перспективы дальнейшей работы: добавить анимацию пошаговой отрисовки с управлением скоростью (ползунок/ввод), реализовать запись последовательности команд для повторного воспроизведения, добавить GUI (Tkinter) для выбора параметров фигур.

Библиографический список

1. Подоксенов М. Н. Опыт проведения вебинаров по математике в рамках подготовки школьников к централизованному тестированию // Материалы XVIII (65) Региональной научно-практической конференции преподавателей, научных сотрудников и аспирантов, Витебск, 13-14 марта 2013 г. : в 2 т. Витебск : ВГУ имени П. М. Машерова, 2013. Т. 2. С. 220-223.
2. Камозина О. В. и др. Веб-приложение «верное решение» для организации олимпиады по математике для учащихся 9 классов // Педагогика. Вопросы теории и практики. 2023. Т. 8. №. 4. С. 374-381.
3. Стукалова Н. А. Повышение качества математической подготовки ориентированных на обучение в вузе старшеклассников в системе дополнительного образования : дис. – Омск : [Ом. гос. пед. ун-т], 2004.
4. Кашина Н. В. Обзор популярных технологий создания электронных учебных пособий // Проблемы управления в социально-экономических и технических системах. 2015. С. 179-183.